US DEPARTMENT OF HOUSING AND URBAN DEVELOPMENT

SINGLE FAMILY HOUSING

FHA CONNECTION

BUSINESS TO GOVERNMENT (B2G)

USER'S GUIDE

SYSTEM	FHA CONNECTION
SUBSYSTEM	B2G
DOCUMENT TYPE	USER'S GUIDE
DOCUMENT DATE	AUGUST 30, 2007

Introduction

The transaction of business with FHA over the Internet using the FHA Connection is a standalone application. It requires the lender to use an Internet browser to establish and maintain an interactive dialogue with an FHA automated system. HUD recognizes that this process has in some cases imposed a burden on mortgagees to work with an additional system, resulting in a duplicity of data entry and processing work. To alleviate this burden, HUD has developed a specification to support integration of often used FHA business functions into existing lender's loan processing systems. Implementation of this specification, named FHA Connection (FHAC) Business-to-Government (B2G) will reduce the FHA loan processing burden for lenders and improve the accuracy of the data submitted through the FHA Connection.

Purpose of FHAC B2G

The FHAC B2G interface specification document provides the necessary information to establish interfaces between the FHA Connection and the lender's loan origination system. This document is primarily aimed at the software development and maintenance staff supporting the lender's loan origination system. This staff may be the lender's IT staff or the staff of a vendor providing the software to the lender. The document contains the following sections:

- Section 1 - Introduction to FHAC B2G includes an overview of FHA B2G.
- Section 2 - Getting Started describes communication protocols, tips on building your B2G interface, and security requirements.
- Section 3 - Interface File Descriptions details XML file formats and error handling.
- Appendices - include a glossary, sample Request and Response Files, and sample Visual Basic source code.

Benefits of FHAC B2G

FHA developed the FHAC B2G to provide lenders with a streamlined process for obtaining FHA underwriting data. Information is sent directly from the lender's loan origination system to the FHA Connection. Some of the benefits achieved by using the FHAC B2G are:

☑ Eliminates duplicate data entry.

☑ Minimizes validation edits.

☑ Easy to retrieve, display, and extract report data.

☑ Standardized tag names can be used for more advanced programming features.

☑ Flexible code that is easily enhanced or modified.

☑ Saves time and resources.

Overview of FHAC B2G

FHAC B2G is a transaction-based system. The steps below describe the processing of loan information from the lender's loan origination system, to the FHA Connection, and back to the lender. These steps must be performed for each transaction processed through FHAC B2G. Detailed information about each available FHAC B2G transaction is available at http://www.hud.gov/offices/hsg/sfh/f17c/welcome.cfm.

1. Loan information is entered into the lender's loan origination system.

2. The lender requests FHA underwriting and electronically sends data to FHAC B2G.

3. FHAC B2G submits loan information directly into the FHA Connection.

4. The FHA Connection processes the information and returns data to FHAC B2G.

5. FHAC B2G feeds data back into the lender's loan origination system.

6. The lender completes loan processing.

Feedback

FHA is committed to providing its lenders with a quality product. Comments provided by the lenders are taken into consideration as enhancements and updates are made to the system. If you have any comments, questions, or suggestions, please contact the FHA Connection Help Line by sending an email message to *sfadmin@hud.gov*. Please include your name and phone number so that you can be contacted in case the FHA requires more information regarding your suggested enhancement.

NOTES:

Getting Started

This section is designed to assist FHAC B2G developers (management and staff) in planning the organization's interface to their loan origination system (LOS). This information is applicable whether you are designing an FHAC B2G interface for the first time or you are updating the existing interface to accommodate a new release.

The following topics are outlined in this section:

- **Things to Consider** - There are decisions that the LOS developer must take into consideration when developing the FHAC B2G interface. While some considerations directly apply to the LOS developers developing the FHAC B2G interface for the first time, other considerations apply to all LOS developers. For this reason, you are encouraged to read this section carefully.

- **Building the FHAC B2G Interface** - There are steps that the LOS developer must follow with regard to implementation of these requirements, whether it be for initial or subsequent use. Most steps are likely to already be a part of your system development process.

- **Data Validation** - There are tasks the LOS Interface Verification Process should accomplish. Most importantly, these tasks center on the issue of data testing to ensure that it is property captured and displayed.

- **Security Requirements and Recommendations** - There are general rules and requirements that apply to the LOS developer when interfacing with FHAC B2G which directly impact system and data-related security.

Things to Consider

This section provides some key points for your consideration in the planning and development of the FHAC B2G interface. These considerations will assist you in analyzing the functionality provided through FHAC B2G and the requirements of the LOS developer.

Interface Data File Formats

FHAC B2G uses Extensible Markup Language (XML). XML uses tags that define the data elements. FHAC B2G uses the MISMO version 1.0.1 DTD with some modifications because the MISMO version 1.0.1 DTD does not define all data elements that are necessary to do business with the FHA. FHAC B2G uses an SSL secured Hypertext Transfer Protocol (HTTPS) Uniform Resource Locator (URL). For further information regarding interface file descriptions, refer to section 3 of this manual. For a current DTD please go to http://www.hud.gov/offices/hsg/sfh/f17c/hud_mismo.dtd.

Communications Protocol

The communications protocol for FHAC B2G is TCP/IP via the Internet. FHA mandates the use of TCP/IP as the communications protocol for all newly developed FHAC B2G interfaces.

Communications Software

When interfacing with FHAC B2G, custom programming provided by the LOS developer is used as the communication software. Most development tools, such as Visual Basic, have HTTPS functionality built within. There are also third party ActiveX controls available.

Integrating FHA B2G with Your LOS

FHA encourages building a seamless interface from the LOS to FHAC B2G. The user interface is the most visible measurement of successful integration because it directly impacts the user's perception of the FHAC B2G service. There are three user interface integration questions which apply to LOS developers with new and existing interfaces to FHAC B2G:

- How do you integrate FHAC B2G request data with existing screens?
- What response data do you extract and store?
- How do you integrate FHAC B2G response data with your LOS?

The majority of the FHAC B2G request data is currently captured by your system because it is standard mortgage application information. However, additional data requests may be required for FHAC B2G. This data should be considered for integration with existing loan processing screens. Your organization is in the position to determine whether it will modify existing screens or it will develop new ones to support additional request data.

Request Functions

The LOS programmer must develop functions to post the data to B2G. For further function specifications, refer to appendix C of this manual.

Extracting and Storing Response Data

Due to the large amount of new data, you should review the Response File data to determine what information you want to extract and store. For detailed information about what information is processed for each transaction, please go to our web site at http://www.hud.gov/offices/hsg/sfh/f17c/welcome.cfm.

Building the FHAC B2G Interface

Data Mapping

The first aspect in developing of the FHAC B2G interface is reviewing the data and formats required by FHAC B2G. These fields are on our web site at http://www.hud.gov/offices/hsg/sfh/f17c/welcome.cfm.

The LOS developer must map each of the FHAC B2G XML data elements to the corresponding data element(s) in the LOS. A field-by-field matrix is the suggested method for accomplishing this data mapping.

Through the data mapping process you are able to identify:

- **Data elements that are currently available within your system** - This set of data elements should not require a significant effort in the process of developing the interface.

- **Data elements that are available but values must be re-mapped or calculated** - These data elements require re-mapping of the values within your system to the values that FHAC B2G requires.

- **Data elements that are unavailable in your system** - The LOS must be modified to capture these additional data elements, utilizing new or existing screens.

LOS Interface Verification

While each organization is ultimately responsible for the testing of its own interface, comments or questions regarding FHAC B2G may be e-mailed to the FHA Connection Help Line at *sfadmin@hud.gov*. The following information should be included within your message to insure that your comments or questions are processed in a timely manner:

- Clearly state your FHAC B2G -related problem(s); and

- List a contact name and telephone number that may be referenced by the help desk personnel.

Note: *The test environment for FHA B2G is* https://www7.hud.gov/b2g/chums*. A user ID and password will be provided to all LOS developers implementing FHAC B2G. If you need a user ID, please send an email requesting an B2G development user ID to* sfadmin@hud.gov*. For a list of all available functions, please go to http://www.hud.gov/offices/hsg/sfh/f17c/welcome.cfm).*

Data Validation

Each LOS must determine the amount of data validation that it will perform. While it is tempting for the LOS to impose edits to ensure that the users are properly entering information, these edits are not required because FHAC B2G will perform all necessary data validation. In an effort to keep all LOS developers informed of any upcoming changes, we keep a list of B2G contacts. If you would like to be added to this list, please send an email to sfadmin@hud.gov. Please note, sometimes the rules change quickly and little notice will be given. Any edits imposed by the LOS are done so at the risk that these edits might prevent their users from performing legitimate FHA functions.

Security Requirements and Recommendations

FHAC B2G's security requirements and recommendations adhere to the standards that have been established by the FHA Connection. All FHAC B2G users are required to have the following:

- **User ID** - six alphanumeric characters; and
- **Password** - six to eight alphanumeric characters.

User IDs and passwords are sent as part of the post in one of the following ways:

- Included in the URL as *https://userid:password@www7.hud.gov/b2g/chums/f17a15c1.cfm*
- Passed as parameters into an ActiveX control used to post https transactions.

Developers needing user IDs for the test environment should send an email to sfadmin@hud.gov. All emails sent to sfadmin@hud.gov should include contact information. Users requiring user IDs and passwords for the production environment are advised to access the FHA Connection at *https://entp.hud.gov/clas/* for further information and direction regarding user registration.

Interface File Descriptions

The loan origination system (LOS) interfacing with FHAC B2G must be able to build B2G standard XML data for a given loan. A new B2G XML Data Stream must be generated for each new request made by the user.

This section details the format of both the XML Data Stream and Response File.

The specific data elements required for each FHA Connection transaction are detailed in the section of the document for that transaction.

XML Data Stream Format

The B2G XML Data Stream uses an SSL secured Hypertext Transfer Protocol (HTTPS) uniform resource locator (URL).

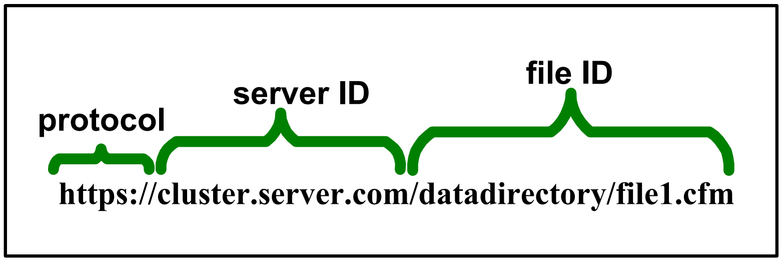

Figure 3.1 - Sections of a URL.

The client sends the XML data stream to the server via a generated post. The XML data stream is comprised of a data element followed by an equal sign (=) and the value of the XML file.

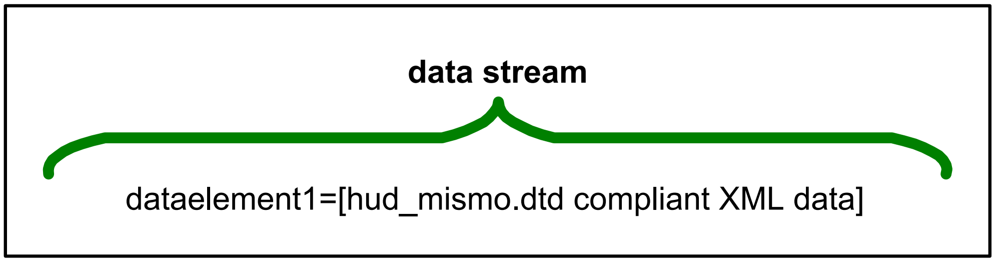

Figure 3.2 - Sample data stream.

The order in which the data elements are listed is not important; however, all data elements required for a given function must be listed. If a required data element is not listed, a processing error is received. All data elements not listed are treated as if the value is blank.

Most B2G functions may be called in either Query or Update mode. In Query mode, the LOS is sending FHAC B2G an FHA Case Number. FHAC B2G then returns the data associated with that particular FHA Case Number. In Update mode, the LOS sends FHAC B2G the necessary data for a transaction. FHAC B2G then edits the data and if there are no errors, processes the proper transaction. The XML tag used to let FHAC B2G know what mode to process the transaction is ProcessStatusCode. The valid values for ProcessStatusCode when calling an FHAC B2G function are:

- Query
- Update

Telecommunications Errors

Since FHAC B2G is using the HTTPS communications protocol, the exact format of telecommunications errors received follows the standard TCP/IP telecommunications errors.

Response File Format

The B2G Response File uses Extensible Markup Language (XML) to identify each data element in the file. XML contains markup symbols to describe the contents of a file. Markup refers to the sequence of characters or other symbols that you insert at certain places in a text file to indicate how the file should look when it is printed or displayed or to describe the document's logical structure.

XML uses a document type definition (DTD) to provide a list of the elements and tags contained in the document. The DTD specifies the structure of the Response File.

Example:

A DTD may say that a <document> contains a title and one or more <book>s, and that every <book> must have exactly one <title>, one or more <author>s, and may or may not contain a single <subtitle>.

The markup indicators are often called "tags." XML tags describe the content in terms of what data is being described. The tags are what define the data elements and are positioned around each data element. The order in which data elements are sent is not significant.

Tags have these characteristics:

- Each tag is a combination of upper- or lowercase letters surrounded by angle brackets (< and >).

- All tags are used in pairs: a start tag and an end tag, which surround the text element.

- End-tags have a forward slash enclosed by angle brackets (</>).

Example:

A <PHONENUM> tag could indicate that the data that followed it was a phone number.
A </PHONENUM> tag indicates the end of the data element.

Processing Errors

FHAC B2G processing errors received are part of the B2G Response File. They are errors that occur while the FHA Connection is processing the request that prevent B2G from returning a completed Response File to the LOS.

The LOS developer can recognize a processing error by examining the ProcessStatusCode. Valid values returned from FHAC B2G for the ProcessStatusCode are:

ProcessStatusCode Value	Description
SuccessUpdatesAllowed	Indicates that the transaction was successfully processed with no errors. A subsequent Update call to this function is allowed.
SuccessNoUpdatesAllowed	Indicates that the transaction was successfully processed with no errors. A subsequent Update call to this function is not allowed. Updates are not allowed on cases that have been endorsed.
Warning	Indicates that the function processed successfully but that at least one warning message is sent. Warning messages are sent in the <ProcessStatusMessage> tag. Each transaction may have multiple <ProcessStatusMessage> tags.
Error	Indicates that an error occurred. The FHA Connection database was not updated by this transaction. Error messages are sent in the <ProcessStatusMessage> tag. Each transaction may have multiple <ProcessStatusMessage> tags.

Example:

```
<ProcessStatusCode>Error</ProcessStatusCode>
<ProcessStatusMessage>SOCIAL SECURITY NUMBER INVALID</ProcessStatusMessage>
```

To receive an error-free Response File, make the necessary corrections and resubmit the transaction. For additional information on correcting errors, contact the FHA Connection Help Line by sending a detailed email message to *sfadmin@hud.gov*.

Telecommunications Errors

Since B2G is using the HTTPS communications protocol, the exact format of telecommunications errors received follows the standard TCP/IP telecommunications errors.

NOTES:

Glossary

B2G Request Data Stream - data sent from the LOS to B2G that contains transaction information.

B2G Response File - data sent from the B2G to LOS that contains response information.

CAIVRS - Credit Alert Interactive Voice Response System; provides the FHA Connection with data on claims, defaults, foreclosures, reviews, or sanctions of potential borrowers and any mortgage credit reject information.

Case Query - allows the user to check the status of a specific case in the loan origination process on the FHA Connection; i.e., to find which stages of processing the case has completed.

document type definition - see *DTD*.

DTD -document type definition; provides a list of the elements, tags, attributes, and entities contained in the document, and their relationship to each other.

Extensible Markup Language - see *XML*.

FHA - Federal Housing Administration; an insuring entity established by legislation.

FHAC - see *FHA Connection*.

FHA case number - unique case identifier sequentially assigned by the FHA Connection during the Case Number Assignment process. Identifies the application for mortgage insurance for a specific property address. It is used throughout the mortgage insurance process to access any FHA Connection function for the specific case. The first three digits indicate the state and Field Office Codes, the next six digits indicate the serial number, and the last digit is a check digit.

FHA Connection - an interactive system on the Internet that gives approved FHA lenders real-time access to FHA systems for the purpose of conducting official FHA business in an electronic fashion.

HTML - HyperText Markup Language; the standard text formatting language of Internet. Internet clients interpret HTML markup and display documents and graphics according to these interpretations. HTML also allows document authors to establish HyperText links between documents in various locations on the Internet, and to create forms interfaces and clickable images that enable users to interact with Web documents.

HTTP - Hypertext Transfer Protocol; the standard method of transferring information across the Internet. HTTP provides a flexible means for communicating a variety of media and file formats across a variety of platforms. This enables users, by working through stable and uncomplicated interfaces, to navigate quickly and easily across a large network of multimedia resources.

HTTPS - Secure HyperText Transfer Protocol.

HUD - U. S. Department of Housing and Urban Development.

hypertext - text within a document that is linked to other text, documents, images, audio, or movies.

lenders - for the purpose of this document, the term lenders represents companies performing the functions required for the process of completing a mortgage transaction.

loan origination software - see *LOS*.

loan origination software developer - for the purpose of this document, the term loan origination software developer indicates the person responsible for making programming changes to the loan origination software.

LOS - loan origination software; for the purpose of this document, the term loan origination software (LOS) indicates software that interfaces loan information to and from FHAC B2G. This applies to commercially-available packages for loan origination used by lenders and lender in-house (customized) software.

processing errors - errors that occur while the FHA Connection is processing the B2G request that prevent the FHA from returning a completed Response File to the LOS.

Request Data Stream- see *B2G Request Data Stream*.

Response File - see *B2G Response File*.

secure socket layer - see *SSL*.

SSL - Secure Socket Layer; incorporates data encryption to protect data transmitted between a B2G user's terminal and the HUD Internet server. SSL is the open standard for secure transactions on the World Wide Web.

tags - markup symbols to describe the contents of a file.

TCP/IP - Transmission Control Protocol/Internet Protocol; the suite of communications protocols used to connect hosts on the Internet. TCP/IP uses several protocols, the two main ones being TCP and IP. TCP/IP is built into the UNIX operating system and is used by the Internet, making it the de facto standard for transmitting data over networks.

telecommunications errors - errors that occur on the Internet that prevent the FHA Connection from receiving the Request Data Stream, or the LOS from receiving the Response File.

Transmission Control Protocol/Internet Protocol - see *TCP/IP*.

uniform resource locator - see *URL*.

URL - uniform resource locator; a standard for identifying objects on the Internet.

XML - Extensible Markup Language; a flexible way to create common information formats and share both the format and the data on the Internet, intranets, and elsewhere.

Sample Files

B2G Request XML Sample

The following B2G Request Data XML example illustrates data that will be sent to FHAC B2G from the lender's loan origination system.

Example: Address Validation Request XML Sample

The following is a sample data stream to be posted in an Address Validation request:

```
InputData=<?xml version='1.0' standalone='yes'?>
<MORTGAGEDATA MISMOVersionID='1.0.1'>
  <PROCESSSTATUS>
          <ProcessStatusRequestor>SAMPLE LOS</ProcessStatusRequestor>
          <ProcessStatusCode>Query</ProcessStatusCode>
  </PROCESSSTATUS>
  <APPLICATION LoanPurposeType = 'Purchase'>
          <AgencyCaseIdentifier>0612345678</AgencyCaseIdentifier>
          <LenderCaseIdentifier>sd-jik3838</LenderCaseIdentifier>
  </APPLICATION>
  <PROPERTY PriorPropertyDisposition = 'n' PROPERTYID = '1' PropertyType = 'Subject'>
          <city>Inwood</city>
          <CountyCode>003</CountyCode>
          <state>WV</state>
          <PostalCode>25428</PostalCode>
          <PARSEDSTREETNAME PARSEDSTREETNAMEID='1'>
                    <HouseNumber>52</HouseNumber>
                    <DirectionPrefix/>
                    <StreetName>Jubal Early</StreetName>
                    <StreetSuffix>AVE</StreetSuffix>
                    <DirectionSuffix/>
                    <ApartmentOrUnit/>
          </PARSEDSTREETNAME>
  </PROPERTY>
  <SUBJECTPROPERTY>
          <CountyCode>003</CountyCode>
          <FinancedNumberOfUnits>1</FinancedNumberOfUnits>
  </SUBJECTPROPERTY>
</MORTGAGEDATA>
```

B2G Response File Samples

The following examples illustrate data that will be returned from FHAC B2G to the lender's loan origination system.

Example 1: Address Validation Response File- No Errors

The following is a sample response file that is error-free:

```
<?xml version='1.0' standalone='yes'?> <MORTGAGEDATA MISMOVersionID='1.0.1'>
  <PROPERTY PROPERTYID='1'>
            <city>INWOOD</city>
            <state>WV</state>
            <PostalCode>254280000</PostalCode>
            <CountyCode>003</CountyCode>
            <PARSEDSTREETNAME PARSEDSTREETNAMEID='1'>
                        <HouseNumber>52</HouseNumber>
                        <StreetName>Jubal Early</StreetName>
                        <StreetSuffix>AVE</StreetSuffix>
            </PARSEDSTREETNAME>
  </PROPERTY>
  <PROCESSSTATUS>
            <ProcessStatusCode>Success</ProcessStatusCode>
            <ProcessStatusMessage sCode='1'>100% Overall probable correctness
</ProcessStatusMessage>

  </PROCESSSTATUS>
</MORTGAGEDATA>
```

Example 2: Address Validation Response File- Errors

The following is a sample response file that has received errors:

```
<?xml version='1.0' standalone='yes'?> <MORTGAGEDATA MISMOVersionID='1.0.1'>
  <PROPERTY PROPERTYID='1'>
            <PostalCode>254280000</PostalCode>
            <PARSEDSTREETNAME PARSEDSTREETNAMEID='1'>

            </PARSEDSTREETNAME>
  </PROPERTY>
  <PROCESSSTATUS>
            <ProcessStatusCode>Error</ProcessStatusCode>
            <ProcessStatusMessage sCode='1'>House # not found on this street
</ProcessStatusMessage>

  </PROCESSSTATUS>
```

```
        </MORTGAGEDATA>
```

Code Samples

Visual Basic Source Code

The following examples illustrate Visual Basic programming code that can be used to develop your FHAC B2G interface. This sample application may be downloaded at http://www.hud.gov/offices/hsg/sfh/f17c/welcome.cfm.

Example 1: Posting a Request Data Stream

The following is sample source code for posting a Case Query or CAIVRS request:

```
Private Sub m_PostData_Click()
  On Error GoTo 0
  Dim sPostData As String
  Dim sProtocol As String
  Dim sURL As String
  Dim iStart As Integer
  Form1.txtData.Text = ""
  If UCase(Mid(Form1.txtURL.Text, 1, 8)) = "HTTPS://" Then
    sProtocol = "https"
    iStart = 9
  ElseIf UCase(Mid(Form1.txtURL.Text, 1, 7)) = "HTTP://" Then
    sProtocol = "http"
    iStart = 8
  Else
    MsgBox "Protocol not supported"
    Exit Sub
  End If
  sPostData = CreateXML
  Form1.txtData.Text = sPostData
  'Exit Sub
  If sPostData = "error" Then
    Exit Sub
  End If
  sURL = sProtocol + "://" + Form1.txtUserID.Text + ":" + Form1.txtPassword.Text + "@" +
Mid(Form1.txtURL.Text, iStart, 1000)
  Inet1.Execute sURL, "POST", sPostData, "content-type:application/x-www-form-urlencoded"
  Do While Inet1.StillExecuting
    DoEvents
  Loop
End Sub



Function CreateXML() As String
CreateXML = "error"
'Open "c:\vbxmlsender\xml.txt" For Binary As #1
```

```
On Error GoTo errhandler
Open Form1.txtFile.Text For Binary As #1
   MyString = "inputData="
   'MyString = "textIn="
   Do While Not EOF(1)
      tmpString = Input(1, #1)
      If tmpString = " " Then
         tmpString = "+"
      End If
      MyString = MyString + tmpString
   Loop
   Close #1


CreateXML = MyString

Exit Function

errhandler:
   MsgBox "Please enter a valid file name"


End Function
```

Example 2: Retrieving a Response File

The following is sample source code for retrieving a Case Query or CAIVRS Response File:

```
Private Sub Inet1_StateChanged(ByVal State As Integer)

   Dim vtData As Variant ' Data variable.
   Debug.Print State
   Select Case State
   ' ... Other cases not shown.
   Case icError ' 11
      ' In case of error, return ResponseCode and       ' ResponseInfo.
        vtData = Inet1.ResponseCode & ":" & _
            Inet1.ResponseInfo
   Case icResponseCompleted   ' 12
        Dim strData As String
        Dim bDone As Boolean: bDone = False
      ' Get first chunk.
        On Error Resume Next
        vtData = Inet1.GetChunk(1024, icString)
        DoEvents
        Do While Not bDone
            strData = strData & vtData
            ' Get next chunk.
            vtData = Inet1.GetChunk(1024, icString)
            DoEvents
            If Len(vtData) = 0 Then
                bDone = True
            End If
        Loop
        On Error GoTo 0
        'txtData.Text = txtData.Text + Chr(13) + Chr(10) + Replace(strData, Chr(10), Chr(13) +
Chr(10), 1, -1)
        'Print #1, Replace(strData, Chr(10), Chr(13) + Chr(10), 1, -1)
        Form1.txtData.Text = Replace(strData, Chr(10), Chr(13) + Chr(10), 1, -1)
        ParseXML (Mid(Form1.txtData.Text, InStr(1, Form1.txtData.Text, "<", vbTextCompare),
Len(Form1.txtData.Text)))

    End Select
End Sub
```

Example 3: Parsing a Response File

The following is sample source code for parsing a Response File. This sample code is using the Microsoft Version 3.0 XML Parser. This is not a recommendation of any particular parser. It is merely meant as a sample to illustrate how to extract data from an XML file. Each LOS is free to choose how they want to parse the returning XML data.

```
Sub ParseXML(sXML As String)
    If Not (MyDoc.loadXML(sXML)) Then
        MsgBox MyDoc.parseError.reason
        Form1.txt_field1.Text = ""
        Form1.txt_field2.Text = ""
        Form1.txt_field3.Text = ""
        Form1.txt_field4.Text = ""
    End If

    If Form1.Combo1.Text = "Address Validation" Then
        Form1.txt_field1.Text = ExtractXMLData("HouseNumber")
        Form1.txt_field2.Text = ExtractXMLData("StreetName")
        Form1.txt_field3.Text = ExtractXMLData("city")
        Form1.txt_field4.Text = ExtractXMLData("state")
    ElseIf Form1.Combo1.Text = "CAIVRS" Then
        Form1.txt_field1.Text = ExtractXMLData("CAIVRSAUTHORIZATIONCODE",
"BORROWER", "ID", "1")
        Form1.txt_field2.Text = ExtractXMLData("CAIVRSAUTHORIZATIONCODE",
"BORROWER", "ID", "2")
        Form1.txt_field3.Text = ExtractXMLData("CAIVRSAUTHORIZATIONCODE",
"BORROWER", "ID", "3")
    End If

End Sub
```

```
Function ExtractXMLData(ElementName As String, Optional ParentName As String, Optional
ParentAttribName As String, Optional ParentAttribValue As String) As String
   Dim objChildNodeList As IXMLDOMNodeList
   Dim objChildNode As IXMLDOMNode
   ExtractXMLData = ""
   If ParentName = "" Then
      Set objNodeList = MyDoc.getElementsByTagName(ElementName)
      If objNodeList.length > 1 Then
         ExtractXMLData = "Error too many elements"
         Exit Function
      End If
      If objNodeList.length = 0 Then
         Exit Function
      End If
      Set objNode = objNodeList.nextNode
      ExtractXMLData = objNode.Text
      Exit Function
   End If

   Set objNodeList = MyDoc.getElementsByTagName(ParentName)
   If objNodeList.length = 0 Then
      Exit Function
   End If
   For x = 1 To objNodeList.length
      Set objNode = objNodeList.nextNode
      For y = 0 To objNode.Attributes.length - 1
         If (objNode.Attributes(y).nodeName = ParentAttribName And
objNode.Attributes(y).nodeValue = ParentAttribValue) Then
            Set objChildNodeList = objNode.childNodes
            For z = 1 To objChildNodeList.length
               Set objChildNode = objChildNodeList.nextNode
               If objChildNode.baseName = ElementName Then
                  ExtractXMLData = objChildNode.Text
               End If
            Next z
         End If
      Next y
   Next x


End Function
```

NOTES: